Efficient Toxic Content Detection by Bootstrapping and Distilling Large Language Models

Jiang Zhang[1]*, **Qiong Wu**[2], **Yiming Xu**[2], **Cheng Cao**[2], **Zheng Du**[2], **Konstantinos Psounis**[1]
[1]University of Southern California, Los Angeles, CA, USA
[2]Amazon.com, Inc., Bellevue, WA, USA
{jiangzha,kpsounis}@usc.edu, {qionggwu,ymxu,chengcao,zhengdu}@amazon.com

Abstract

Toxic content detection is crucial for online services to remove inappropriate content that violates community standards. To automate the detection process, prior works have proposed varieties of machine learning (ML) approaches to train Language Models (LMs) for toxic content detection. However, both their accuracy and transferability across datasets are limited. Recently, Large Language Models (LLMs) have shown promise in toxic content detection due to their superior zero-shot and few-shot in-context learning ability as well as broad transferability on ML tasks. However, efficiently designing prompts for LLMs remains challenging. Moreover, the high run-time cost of LLMs may hinder their deployments in production. To address these challenges, in this work, we propose **BD-LLM**, a novel and efficient approach to **B**ootstrapping and **D**istilling **LLM**s for toxic content detection. Specifically, we design a novel prompting method named Decision-Tree-of-Thought (DToT) to bootstrap LLMs' detection performance and extract high-quality rationales. DToT can automatically select more fine-grained context to re-prompt LLMs when their responses lack confidence. Additionally, we use the rationales extracted via DToT to fine-tune student LMs. Our experimental results on various datasets demonstrate that DToT can improve the accuracy of LLMs by up to 4.6%. Furthermore, student LMs fine-tuned with rationales extracted via DToT outperform baselines on all datasets with up to 16.9% accuracy improvement, while being more than 60× smaller than conventional LLMs. Finally, we observe that student LMs fine-tuned with rationales exhibit better cross-dataset transferability.

Introduction

Toxic content detection is important for online services to protect users from harmful and offensive content, ensuring a safer and more positive user experience. Common toxic content categories include hate speech, biased content, sexual content, violent content, bullying content, etc. Due to the massive amount of content on the Internet, it is impractical to manually check the toxicity of each content. Hence, machine learning (ML) solutions based on supervised learning have been widely applied to automate the toxic content detection process, where Language Models (LMs) fine-tuned on a task-specific dataset achieve the state-of-the-art (SOTA) performance (Caselli et al. 2020; Kim, Park, and Han 2022).

However, existing supervised learning ML solutions face three challenges. First, they require training data with labels, which are non-trivial to obtain for toxic content detection tasks due to the lack of standard definitions, especially for implicit toxic content. Second, the fine-tuned LMs may overfit the training dataset, which limits the transferability to other datasets. Lastly, they typically can only predict binary labels without detailed reasoning.

To handle the above challenges, the recently emerging Large Language Models (LLMs) have been leveraged to detect toxic content (Wang and Chang 2022; Zhang et al. 2023), due to their superior zero-shot and few-shot in-context learning performance and transferability. Existing works on LLMs for toxic content detection focus on designing novel prompting approaches to enhance the performance of LLMs. However, their performance relies heavily on the quality of prompts, which are non-trivial to design. Moreover, deploying LLMs for toxic content detection in production can incur both high run-time cost and high latency, especially when the number of tokens in the prompt is large (e.g. for in-context few-shot learning).

Motivated by the above, in this paper, we propose **BD-LLM**, a novel and efficient approach to **B**ootstrapping and **D**istilling **LLM**s for toxic content detection (as shown in Figure 1). We first design a novel prompting approach called Decision-Tree-of-Thought (DToT) to improve the zero-shot and few-shot in-context learning performance of LLMs as well as extract better rationales. At a high level, DToT works by iteratively selecting more fine-grained context to re-prompt LLMs whenever they have low-confident answers. It automatically decides when to select more fine-grained context for re-prompting and which type of fine-grained context should be selected. Second, we propose to fine-tune a student LM with smaller model size to predict both labels and rationales extracted via DToT.

We evaluate the proposed approach on three public datasets and one Amazon internal dataset. Experimental results demonstrate that employing DToT prompting consistently leads to improved zero-shot and few-shot in-context learning performance on all datasets, by up to 4.6% higher accuracy. Furthermore, we demonstrate that fine-tuning student LMs with DToT-extracted rationales results in up

*This work was done when the author was an intern at Amazon.

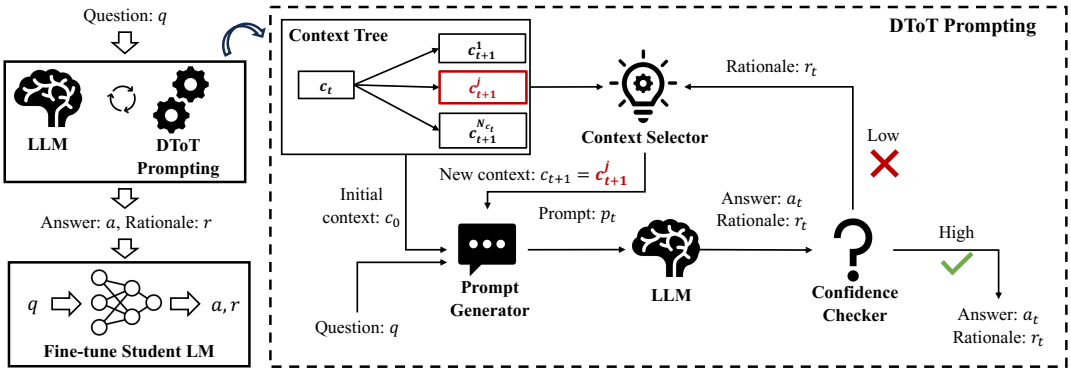

Figure 1: Overall workflow of the proposed BD-LLM. Given question q, it first bootstraps the LLM via DToT prompting to extract answer a and rationale r with high-confidence. Then, it uses q as input and (a, r) as output to fine-tune the student LM.

to 16.9% accuracy improvement compared with baselines. Meanwhile, these student LMs have model size more than 60× smaller than conventional LLMs. Finally, we observe that the cross-dataset transferability of student LMs fine-tuned with rationale is significantly improved.

Our contributions are summarised as follows:

- We propose BD-LLM, an efficient approach for toxic content detection, which leverages LLMs strength but also reduces their complexity via bootstrapping and distillation. This is the first-of-its-kind study on toxic content detection.
- To bootstrap LLMs, we design a novel prompting method named DToT, which selectively re-prompts LLMs with more fine-grained context to boost their detection performance and extract high-quality rationales.
- To distill LLMs into smaller student LMs, we fine-tune the student LMs to predict both labels and rationales extracted via DToT.
- We evaluate the proposed solution on four datasets and demonstrate that DToT can improve the toxic content detection accuracy of LLMs by up to 4.6% and student LMs fine-tuned with DToT-extracted rationales achieve the SOTA performance with up to 16.9% accuracy improvements and more than 60× smaller size.

Related Work

Toxic Content Detection

Prior works on toxic content detection can be categorized into two types. One type of research works focuses on creating benchmark datasets for toxic content detection, either by crowdsourcing and annotating human-written text (Ye, Le, and Lee 2023; Sap et al. 2019; Vidgen et al. 2020), or leveraging ML-based approaches to generate high-quality toxic dataset in a scalable way (Hartvigsen et al. 2022). Another type of works proposes novel approaches to fine-tune LMs on toxic dataset. Caselli et al. (2020) propose HateBERT, a pre-trained BERT model for abusive language detection, which significantly outperforms the original BERT model. Kim, Park, and Han (2022) propose a novel contrastive learning method to improve the cross-dataset performance of HateBert. Most recently, researchers have started to use LLMs to detect toxic content. Wang and Chang (2022) design a generative prompt-based inference method to leverage LLMs for toxic content detection. Zhang et al. (2023) propose an interpretable, unified, language checking method (UniLC) to enhance the few-shot in-context learning capabilities of LLMs for misinformation, stereotypes, and hate speech detection. Compared with these works, our work not only proposes a novel and orthogonal prompting approach that improves the zero-shot/few-shot performance, but also extracts and distills rationales into a smaller but more effective student LM for toxic context detection.

Prompting LLMs

LLMs such as GhatGPT (OpenAI 2023) and Llama (Touvron et al. 2023) have demonstrated superior zero-shot/few-shot in-context learning capabilities and generalizability on a variety of language tasks without fine-tuning (Kojima et al. 2022). However, their performance is heavily related to the quality of input prompts (Zhao et al. 2023). Hence, varieties of prompting approaches have been proposed to improve the quality and robustness of LLMs' response (Wei et al. 2022; Yao et al. 2023; Wang et al. 2022b; Luo et al. 2023; Wang, Zhu, and Wang 2023; Chen et al. 2023). Wei et al.(2022) and Wang et al.(2022b) propose Chain-of-Thought (CoT) and self-consistent CoT, which prompts LLMs step by step to improve LLMs' performance on reasoning tasks. Yao et al.(2023) generalizes CoT into Tree-of-Thought (ToT), which enables LLMs to explore multiple intermediate steps for complex problem solving tasks. Different from CoT and ToT which are suitable for step-by-step reasoning problems, the proposed DToT in this work is designed for classification problems with a novel confidence checker and context selector, which iteratively searches and injects more fine-grained context into prompts to enhance the classification confidence of LLMs. Other works have proposed to leverage the in-context learning capability of LLMs, by augmenting the prompts with demonstrations (Wang, Zhu, and Wang 2023), rationales (Chen et al. 2023) or grounded information (Luo et al. 2023; Zhang et al. 2023). Note that our DToT prompting is orthogonal to existing in-context learn-

ing methods for LLMs (see Section 3.2).

Distilling LLMs

Some recent works also tackle the problem by distilling LLMs into smaller LMs for domain-specific tasks (Magister et al. 2022; Hsieh et al. 2023; Wang et al. 2022a, 2023). Wang et al. (2022a) propose PINTO, which uses LLMs' rationales as context of a small LM to improve its performance on reasoning tasks. Wang et al. (2023) further propose SCOTT, a faithful knowledge distillation method that elicits self-consistent rationales from LLMs to fine-tune a small while more faithful LM for open-domain question-answering tasks. Both Magister et al. (2022) and Hsieh et al. (2023) demonstrate that LLMs can be distilled into smaller but more effective LMs by fine-tuning with both answers and rationales on commonsense reasoning and arithmetic tasks. Different from these works, our work focus on the domain of toxic content detection. Moreover, we propose DToT that extracts better rationales from LLMs and demonstrate that using DToT-extracted rationales further improves the effectiveness of distillation for toxic content detection.

Approach

Figure 1 demonstrates the overall workflow of BD-LLM, which consists of two separate steps. Specifically, in the first step, we design DToT prompting to iteratively extract high-confidence answers a and rationales r from LLM given input question q. [1] In the second step, we conduct rationale distillation by fine-tuning a student LM to generate both a and r given input q. We describe the details of each step below.

DToT Prompting

At a high level, DToT prompting iteratively selects more fine-grained context and re-prompts the LLM when they output responses with low confidence. Two challenges in designing DToT prompting are: *1) how to decide whether the response of LLM is confident or not*; *2) how to decide which type of fine-grained context should be selected to re-prompt the LLM*[2]. To address the first challenge, we design a confidence checker to measure the self-confidence score of the LLM's answer. To handle the second challenge, we design a context selector to select appropriate fine-grained context from a context tree based on the LLM's rationale.

In total, DToT prompting consists of four modules: 1) confidence checker; 2) context tree; 3) context selector; 4) prompt generator, which can be used for both *black-box* and *white-box* LLMs. Note that for *black-box* LLMs, we assume that we only have access to their output responses (e.g. ChatGPT). By contrast, for *white-box* LLMs, we assume that we can also have access to their model parameters (e.g. open-source LLMs). The detailed design of these modules and the end-to-end workflow are presented below.

[1] Note that for toxic content detection, an answer is either 'Yes' or 'No', while a rationale consists of one or more sentences.

[2] Since there are many different types of toxic content and for each of them we need to use the right context (i.e. part of the prompt that specifies the criteria to call certain content as toxic), there is a need to automatically select the appropriate context for prompting.

Confidence Checker. This module is designed to measure the self-confidence score of LLM's response, which is defined as s_t^{conf} for iteration step t . Specifically, for the *black-box* LLM, the confidence checker uses the toxicity rating of LLM (defined as $s_t^{toxi} \in [0, 100]$) to calculate the confidence score. If s_t^{toxi} is above a maximal threshold s_h or below a minimal threshold s_l, we consider the answer as confident, vice versa. Note that to obtain s_t^{toxi}, we explicitly require the LLM to output the toxicity rating in the prompt p_t. For the *white-box* LLM (e.g. any open-source LLMs), the confidence checker will leverage the output logits of LLM to calculate the probability of generating answers a_t conditional on the prompt p_t. This probability will be used as the confidence measurement (i.e. $s_t^{conf} = \Pr[a_t|p_t]$). Formally, we define s_t^{conf} as:

$$s_t^{conf} = \begin{cases} \mathbf{1}[s_t^{toxi} \notin (s_l, s_h)], & \text{for } \textit{black-box} \text{ LLMs} \\ \Pr[a_t|p_t], & \text{for } \textit{white-box} \text{ LLMs} \end{cases} \quad (1)$$

where $\mathbf{1}[\cdot]$ is an indicator function, and s_l and s_h are two adjustable thresholds (see Appendix A for selecting s_l, s_h).

Suppose the confidence checker has measured the self-confidence score s_t^{conf} of LLM's answer a_t. Then, if s_t^{conf} is higher than a threshold s_δ, the checker will consider a_t as a confident answer. Otherwise, a_t is considered to be unconfident. Formally, the output of the confidence checker D_{check} is defined as:

$$D_{check}(s_t^{conf}) = \begin{cases} \text{Unconfident}, & \text{if } s_t^{conf} \in [0, s_\delta) \\ \text{Confident}, & \text{if } s_t^{conf} \in [s_\delta, 1] \end{cases} \quad (2)$$

where s_δ is an adjustable threshold (see Appendix A for selecting s_δ).

Context Tree. Before introducing the context selector module, we first provide the definition of context tree. Suppose the universe of context is represented as set $\mathcal{C}$. We define the context tree as $T_c : \mathcal{C} \to \textsc{List}[\mathcal{C}]$, which is a mapping from a parent-node context $c_t \in \mathcal{C}$ to the list of its child-node contexts $[c_{t+1}^1, ..., c_{t+1}^{N_{c_t}}]$, where N_{c_t} is the total number of child nodes of c_t, and $c_{t+1}^j \in \mathcal{C}$ is the j-th child-node context of c_t. Moreover, each child-node context's category is designed to be a subcategory of its parent-node context's category, such that the child-node context is more fine-grained than parent-node context. For instance, suppose that c_0 provides the definition of toxic content, which includes hate speech, sexual content, etc. Then, c_1^1 can be a more fine-grained definition for hate speech, and c_1^2 can be a more fine-grained definition for biased content (see Figure 2).

Context Selector. Now we introduce the context selector module, which takes as input the rationale r_t and the context c_t in prompt at step t, and a context tree T_c, to select a more fine-grained context c_{t+1} for step $t + 1$. Whenever an answer a_t at iteration step t is considered as unconfident by the confidence checker, the context selector will select new context from the context tree for re-prompting LLMs. Specifically, it measures the relevance between rationale r_t and each child-node context c_{t+1}^j, and then selects the most relevant one $c_{t+1} = c_{t+1}^{j^*}$. Formally, we define the output of

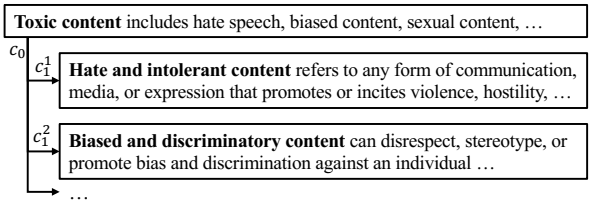

Figure 2: An example of context tree.

context selector as:

$$D_{select}(r_t) = j^* = \arg\max_{1 \le j \le N_{c_t}} s_t^{rele,j}(r_t, c_{t+1}^j), \quad (3)$$

where $[c_{t+1}^1, ..., c_{t+1}^{N_{c_t}}] = T_c(c_t)$, and $s_t^{rele,j}$ is the relevance score between r_t and c_{t+1}^j.

Furthermore, to calculate $s_t^{rele,j}$ for LLMs which can generate relatively high-quality rationales (mostly for *black-box* LLMs like ChatGPT), we use a classification prompt to ask LLM which of these child-node context categories are most relevant to the rationale r_t. We denote the answer from the LLM as $\mathrm{Class}(r_t)$. By contrast, for *white-box* LLMs which cannot generate rationales with decent quality, we construct a set of candidate rationales as $r_t = [r_t^1, ..., r_t^{N_{c_t}}]$, where each $r_t[j] = r_t^j$ is relevant to only one of the child-node contexts c_{t+1}^j. Since we have access to the output logits of the LLM, we measure the probability of generating each candidate rationale conditional on the prompt p_t (i.e., $\Pr[r_t[j]|p_t]$) as the relevance score. Formally, we define $s_t^{rele,j}$ as:

$$s_t^{rele,j} = \begin{cases} \mathbf{1}[j = Class(r_t)], & \text{for } \textit{black-box} \text{ LLMs} \\ \Pr[r_t[j]|p_t], & \text{for } \textit{white-box} \text{ LLMs} \end{cases} \quad (4)$$

where $\mathbf{1}[\cdot]$ is an indicator function.

Prompt Generator. The Prompt Generator module P generates an input prompt p_t for LLM M based on the question q, and the context c_t at iteration step t. Note that it will modify the question q with the change of context. For example, the initial question based on c_0 is designed to ask whether a statement contains toxic content. Suppose c_1 is selected to provide context related to hate speech. Then, the question will be modified to ask whether the statement contains hate speech, which is a specific category of toxic content. We provide the prompt templates in Appendix C.

End-to-end Workflow. Algorithm 1 illustrates the end-to-end workflow of DToT prompting, where the input contains LLM M, question q, initial context c_0. At each iteration step t (line 2), it first generates prompt p_t via prompt generator P and gets LLM M's output (lines 4-9). Next, it calculates the confidence score of LLMs's answer s_t^{conf} and decides whether it is confident via the confidence checker D_{check} (lines 11-12). For unconfident answers, it will get a list of new candidate contexts from context tree T_c (line 14), calculate the relevance score between LLM's rationale and each one of the new context in the list (lines 15-22), and select the most relevant new context (lines 23-24). Lastly, it terminates when the maximal iteration step T is reached (line 2) or the LLM's answer is confident (line 27).

Algorithm 1: DToT Prompting

Input: Question q, initial context c_0, LLM M, thresholds s_l, s_h, and s_δ, maximal iteration step T.
Output: Answer a, rationale r.

1: Define current step as $t = 0$.
2: **while** $t < T$ **do**
3: // Generate prompt and get response
4: Generate input prompt: $p_t = P(q, c_t, M)$.
5: **if** M is black-box **then**
6: Get LLM output: $(a_t, s_t^{toxi}, r_t) = M(p_t)$.
7: **else**
8: Get LLM output: $(a_t, r_t) = M(p_t)$.
9: **end if**
10: // Check the confidence of answer
11: Calculate confidence score s_t^{conf} by Eq. (1).
12: **if** $D_{check}(s_t^{conf}) = \mathrm{Confidence}$: **then**
13: // Select new context from context tree
14: Get the new context list: $[c_{t+1}^j]_{j=1}^{N_{c_t}} = T_c(c_t)$.
15: **if** M is black-box **then**
16: Let M get the rationale class: $\mathrm{Class}(r_t)$.
17: **else**
18: Let r_t be the candidate rationale list: $[r_{t,j}]_{j=1}^{N_{c_t}}$.
19: Calculate relevance score $[s_t^{rele,j}]_{j=1}^{N_{c_t}}$ by Eq. (4).
20: **end if**
21: Calculate $j^* = D_{select}(r_t)$ by Eq. (3).
22: Set new context c_{t+1} as $c_{t+1}^{j^*}$ and $t = t + 1$.
23: **else**
24: Exit.
25: **end if**
26: **end while**
27: **return** $a = a_t, r = r_t$

Augmented DToT Prompting

So far, we present how DToT prompting works under the zero-shot learning setup. Considering that DToT prompting is orthogonal to few-shot learning, we propose two augmented versions of DToT prompting below.

DToT+FS. The approach combines DToT prompting with the vanilla few-shot in-context learning, which adds a few demonstrations in the prompt. Recent work (Wang, Zhu, and Wang 2023) has shown that a few demonstrations can effectively improve the in-context learning performance of LLMs. Moreover, motivated by prior work (Liu et al. 2021), we select K positive statements and K negative statements that are most semantically similar to the input statement as demonstrations from a development set.

DToT+FS+R. Recent works (Sun et al. 2022; Zhang et al. 2023) have demonstrated that rationales or facts in prompts serve as grounded information to further enhance the few-shot in-context learning capability of LLMs. Therefore, on top of DToT+FS approach, we include the rationale for each demonstration in the prompt.

Rationale Distillation

After obtaining the answers and rationales from LLMs via DToT, we can distill LLMs into smaller student LMs, which

can predict both answers and rationales. We describe the details of rationale distillation under two scenarios below.

Distillation without Labels. Assume that we do not have ground truth labels for training inputs. Hence, we use both the answers and rationales output by LLMs as ground truth. In practice, this approach can be applied when it is challenging or costly to obtain ground truth labels. Suppose for the i-th input x_i, the predicted label from LLM is $\hat{y}_i$ [3] and the associated rationale is r_i. Suppose the student model is $(y_i^s, r_i^s) = f_\theta^s(x_i)$ parameterized by θ. Then the loss function is defined as

$$L^s(\theta) = \sum_{i=1}^{D} (CE(y_i^s, \hat{y}_i) + CE(r_i^s, r_i)), \tag{5}$$

where D is the total number of training data points and CE represents the cross-entropy averaged at token-level.

Distillation with Labels. Assume that we have access to ground truth labels for training inputs. Hence, we can use the ground truth labels for fine-tuning. Specifically, if the LLMs can not predict the right answer, we only use the binary ground truth labels. Otherwise, we use both the labels and rationales. Formally, the loss function is defined as

$$L^s(\theta) = \sum_{i=1}^{D} (CE(y_i^s, y_i) + \lambda * \mathbf{1}_{y_i=\hat{y}_i} CE(r_i^s, r_i)), \tag{6}$$

where λ is an adjustable parameter (see Appendix A for selecting λ).

Experimental Setup

Datasets

We evaluate our approach on three public datasets and an Amazon private dataset.

Toxigen. This is an GPT3-generated dataset provided by Hartvigsen et al. (2022), which contains both toxic and benign statements about 13 minority groups. We use their annotated dataset with 8,960 training statements and 940 testing statements in our experiments, where 40% of statements are toxic. Note that we exclude about 5% ambiguous statements with toxicity level 3 in experiments, where the scores (i.e. toxicity level) range from 1 to 5 and 3 denotes ambiguity, thus are removed from our experiments.

SBIC. This dataset contains 44,671 social media posts from Reddit, Twitter, and hate sites. Each post was annotated by the Social Bias Frames proposed in (Sap et al. 2019) to specify whether there exists any social bias and stereotype towards a target group that is toxic. In our experiments, we randomly and uniformly sample 4,000 statements as training dataset and 1,000 statements as testing dataset, 50% of which are toxic.

DHate. This dataset is generated by a human-and-model-in-the-loop process proposed in (Vidgen et al. 2020). In total, it contains about 40,000 labeled statements, of which 54% contains hate content. We randomly and uniformly sample 4,000 statements as training dataset and 1,000 statements as testing dataset in our experiments.

Amazon. This is a private dataset from Amazon, which contains 8,000 benign statements and 2,000 toxic statements annotated by professional human labelers. This dataset is only used for testing purposes in our experiments due to confidentiality policy.

Models and Baselines

DToT prompting. We evaluate the effectiveness of DToT prompting on both *black-box* and *white-box* models, which are defined in Section 3.1. Specifically, we select gpt-3.5-turbo (denoted by ChatGPT (OpenAI 2023)) with 175B parameters as our *black-box* model, and we consider FastChat-T5 (denoted by FC-T5 (Zheng et al. 2023)) with 3B parameters as our *white-box* model.

Moreover, we compare DToT prompting with three existing baselines: (a) RoBerta model fine-tuned on each dataset, since prior work (Hartvigsen et al. 2022) has shown that it can achieve the SOTA performance on Toxigen dataset. (b) CoT prompting, which can be viewed as a special case of DToT prompting without iteratively re-prompting. (c) UniLC prompting proposed by Zhang et al. (2023).

Finally, we compare DToT prompting with its two augmented versions: DToT+FS and DToT+FS+R (see Section 3.2). Note that for DToT+FS, we select $K = 3$ positive statements and $K = 3$ negative statements that are most semantically similar to the input statement from a development set. To measure the similarity, we use sentence transformer (Reimers and Gurevych 2019) to convert each statement into an embedding, and use the cosine similarity between two statements as a measurement of their semantic similarity. Moreover, for DToT+FS+R, we use the rationales associated with the correct answers generated by ChatGPT as augmentations.

Model Distillation. For our main experiments, we select FC-T5 to evaluate the effectiveness of fine-tuning a student LM using rationales generated from LLMs, and we consider two baseline approaches: (a) fine-tuning with labels without rationale, and (b) fine-tuning with rationales generated by CoT prompting (which we denote by $\mathbf{R}_{CoT}$). Note that for our approach (which we denote by $\mathbf{R}_{DToT}$), we use the rationales generated via DToT+FS+R prompting as the ground truth during fine-tuning. Furthermore, to investigate how the number of parameters in student LM affects the fine-tuning performance, we use Flan-T5 models with different model size (Chung et al. 2022) (see Appendix A for more details on why we select these models and hyperparameter setup).

Evaluation Results

Evaluation of DToT

We start by evaluating DToT prompting to answer the following question.

Q1: Can DToT prompting enhance the detection performance of LLMs? As shown in Table 1, compared with CoT prompting, DToT prompting can enhance the zero-shot learning performance of both *black-box* model and *white-box* model significantly on all three public datasets. Specifically, for FC-T5, DToT prompting can increase the accu-

[3] Note that we assign label 1 for 'Yes' answer and label 0 for 'No' answer.

Model	Method	Toxigen			SBIC			DHate		
		Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC
RoBerta	FT	82.45	75.35	90.24	82.80	83.96	91.82	70.60	75.38	81.27
FC-T5	CoT	79.69	73.49	85.54	63.50	71.15	67.76	63.40	71.09	69.13
	DToT	81.24	75.36	86.67	**68.80**	**74.26**	72.47	**66.80**	**73.05**	72.66
	DToT+FS	81.90	75.88	86.74	68.00	73.77	**72.74**	65.40	72.23	**73.80**
	DToT+FS+R	**82.01**	**75.99**	**86.94**	68.00	73.77	72.56	66.50	72.87	73.37
ChatGPT	CoT	82.71	81.29	N/A	68.50	72.82	N/A	65.20	72.20	N/A
	UniLC	83.30	82.73	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	DToT	85.03	82.76	N/A	71.60	74.18	N/A	68.20	73.92	N/A
	DToT+FS	86.03	83.51	N/A	71.70	74.62	N/A	**69.50**	**74.33**	N/A
	DToT+FS+R	**87.03**	**85.06**	N/A	**72.00**	**74.91**	N/A	69.20	74.30	N/A

Table 1: Evaluation results of DToT on Toxigen, SBIC, DHate datasets. In "Method" column, "FT" stands for fine-tuning on training dataset, "CoT" refers to CoT prompting, "DToT" corresponds to DToT prompting, "DToT+FS" denotes the combination of DToT prompting with few-shot demonstrations, and "DToT+FS+R" presents the combination of DToT prompting with few-shot demonstrations and rationale augmentations. Due to the lack of output logits, the AUC scores of ChatGPT are populated as "N/A".

racy by up to 5.30% and the F1 score by up to 3.11%. It is worth noting that DToT prompting can also improve the AUC score of FC-T5 on all datasets, indicating its robust performance. For ChatGPT, DToT prompting consistently outperforms CoT prompting, and increases the accuracy by up to 3.10% and the F1 score by up to 1.72%.

Moreover, combining DToT with few-shot in-context learning (i.e. DToT+FS and DToT+FS+R) may further improve models' performance. For instance, on Toxigen dataset, compared with the vanilla DToT prompting, adding demonstrations (i.e. DToT+FS) can improve ChatGPT's accuracy by 1.00%. Furthermore, by incorporating both demonstrations and rationales during prompting (i.e. DToT+FS+R), ChatGPT's accuracy can be improved by 2.00% respectively, outperforming baselines by up to 4.58% (for RoBerta) and at least 3.73% (for UniLC).

Therefore, we conclude that DToT prompting and its augmented versions significantly enhance LLMs' performance on toxic content detection.

Evaluation of Rationale Distillation

Next, we evaluate whether fine-tuning with rationales extracted via DToT improves the performance of student LMs. We first answer the following question:

Q2: Can fine-tuning with rationales extracted via DToT derive a student LM with higher accuracy? Table 2 reports the evaluation results of fine-tuning with or without rationales. In this table, "Label = Human" denotes the utilization of the ground truth labels in training datasets for fine-tuning, "Label = LLM" means the usage of labels predicted by the teacher LLM for fine-tuning, "Rationale = N/A" indicates that fine-tuning is conducted solely with labels, "Rationale = $\mathbf{R}_{CoT}$" implies that fine-tuning takes place with both labels and CoT-extracted rationale from teacher LLMs, and "Rationale = $\mathbf{R}_{DToT}$" represents that we fine-tune the model with both labels and DToT-extracted rationales from the teacher LLM (i.e. ChatGPT).

As reported in Table 2, FC-T5 fine-tuned with DToT-extracted rationales and ground truth labels outperforms all baselines across various public datasets. Notably, this fine-tuning approach yields significant improvements in accuracy, F1 score, and AUC score. Specifically, compared with fine-tuning with labels only, it can increase the model accuracy by up to 2.54%, the F1 score by up to 2.46%, and the AUC score by up to 1.94% respectively. Compared with the prior SOTA LM fine-tuned on these datasets (i.e. RoBerta), it can significantly increase the model accuracy by up to 15.80%, the F1 score by up to 11.49%, and the AUC score by up to 13.22% respectively. In addition, FC-T5 fine-tuned with labels and rationales outperforms teacher LLM by up to 16.90% w.r.t. accuracy, with 60× smaller model size.

Moreover, compared with fine-tuning with $\mathbf{R}_{CoT}$ and ground truth labels, we observe that fine-tuning with $\mathbf{R}_{DToT}$ and ground truth labels can consistently result in student LMs with better detection performance on all public datasets. This indicates that rationales generated via DToT prompting ($\mathbf{R}_{DToT}$) have higher quality than those generated via CoT prompting ($\mathbf{R}_{CoT}$).

Lastly, under the scenario where we use labels predicted by teacher LLM as ground truth, we observe that fine-tuning with $\mathbf{R}_{DToT}$ and labels can still outperform both fine-tuning with $\mathbf{R}_{CoT}$ and labels, and fine-tuning with only labels. Specifically, on Toxigen dataset, FC-T5 fine-tuned with $\mathbf{R}_{DToT}$ and predicted labels from teacher LLM can even outperform RoBerta fine-tuned with human labels from training dataset. However, without using $\mathbf{R}_{DToT}$, the fine-tuned FC-T5 cannot outperform Roberta.

In summary, we conclude that fine-tuning with both labels and rationales can effectively improve the student LMs' performance. Moreover, using rationales with better quality (i.e. DToT-extracted rationales versus CoT-extracted rationales) can further enhance the performance of fine-tuned LMs.

Q3: Can fine-tuning with rationales improve the transferability of student LMs across different toxic datasets? To evaluate whether fine-tuning with both labels and rationales can improve the transferability of student LMs, we test the models fine-tuned on the other two public datsaets (SBIC and DHate) and one private dataset (Amazon). Table 3 reports our transferability evaluation results, where we use AUC score as our metric. First, we observe that

Model	Label	Rationale	Toxigen			SBIC			DHate		
			Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC
RoBerta	Human	N/A	82.45	75.35	90.24	82.80	83.96	91.82	70.60	75.38	81.27
ChatGPT	N/A	N/A	87.03	85.06	N/A	72.00	74.91	N/A	69.50	74.33	N/A
FC-T5	LLM	N/A	81.90	80.19	91.78	64.90	71.99	74.67	63.50	71.15	70.50
		$\mathbf{R}_{CoT}$	81.79	80.10	92.88	67.30	73.35	74.25	64.30	71.60	72.90
		$\mathbf{R}_{DToT}$	**84.00**	**82.08**	**93.60**	**69.00**	**74.42**	**81.09**	**68.00**	**73.77**	**77.89**
	Human	N/A	84.99	83.00	92.43	84.00	84.91	92.89	85.00	85.71	93.64
		$\mathbf{R}_{CoT}$	87.31	85.24	94.15	84.20	85.07	93.18	86.20	86.71	93.71
		$\mathbf{R}_{DToT}$	**87.53**	**85.46**	**94.37**	**85.10**	**85.82**	**93.85**	**86.40**	**86.87**	**94.49**

Table 2: Distillation evaluation results on Toxigen, SBIC, and DHate datasets. In "Label" column, "Human" indicates that the labels come from the training dataset, "LLM" indicates that the labels are predicted by LLM. In "Rationale" column, "N/A" means no rationales are used in fine-tuning, "$\mathbf{R}_{CoT}$" means rationales extracted via CoT are used in fine-tuning, and "$\mathbf{R}_{CoT}$" means rationales extracted via DToT are used in fine-tuning.

Model	Labels	Rationale	SBIC	DHate	Amazon
RoBerta	Human	N/A	65.46	61.51	X
ChatGPT	N/A	N/A	72.00	69.50	N/A
FC-T5	LLM	N/A	75.36	74.96	X+2.47
		$\mathbf{R}_{CoT}$	71.48	70.99	X+4.52
		$\mathbf{R}_{DToT}$	**75.90**	**77.31**	**X+4.54**
	Human	N/A	75.24	77.75	X+0.05
		$\mathbf{R}_{CoT}$	77.15	**78.97**	**X+4.16**
		$\mathbf{R}_{DToT}$	**77.29**	77.54	X+3.61

Table 3: Transferability evaluation results. Note that we fine-tune these models on Toxigen dataset while testing them on other datasets, and we report AUC score. For Amazon dataset, due to confidentiality policy, we only report the increased AUC score compared with RoBerta (whose AUC score is denoted by X).

Model	Size	Rationale	Acc	F1	AUC
FC-T5	3B	N/A	84.99	83.00	92.43
		$\mathbf{R}_{DToT}$	**87.53**	**85.46**	**94.37**
F-T5-XL	3B	N/A	85.54	83.52	92.51
		$\mathbf{R}_{DToT}$	87.53	85.42	93.17
F-T5-L	770M	N/A	83.44	81.57	92.42
		$\mathbf{R}_{DToT}$	84.22	82.24	93.18
F-T5-B	220M	N/A	81.02	79.43	91.26
		$\mathbf{R}_{DToT}$	81.35	79.66	91.16

Table 4: Impact of student LMs' size on rationale distillation. Note that we use Toxigen dataset for a case study, and F-T5-XL/F-T5-L/F-T5-B is short for Flan-T5-XL/Flan-T5-Large/Flan-T5-Base.

compared with fine-tuning with labels only (Rationale = N/A), fine-tuning with both labels and rationales (Rationale = $\mathbf{R}_{CoT}$/$\mathbf{R}_{DToT}$) can improve the AUC score of student LMs by up to 2.35% on testing datasets. Second, FC-T5 fined-tuned with both labels and rationales have significantly better AUC scores on all datasets compared with RoBerta. Lastly, while we only fine-tune FC-T5 on Toxigen datset, it still outperforms teacher LLM (i.e. ChatGPT) on both SBIC and DHate datasets. Hence, we conclude that fine-tuning with rationale effectively improves the cross-dataset transferability of student LMs.

Q4: How does student LMs' size affect the detection accuracy? So far, we use FC-T5 with 3B parameters as our student LM during rationale distillation experiments. To investigate how the model size affects the student LMs' performance, we fine-tune Flan-T5 models of different size with rationales on Toxigen dataset. As reported in Table 4, fine-tuning with both labels and rationales can consistently enhance the student LMs' performance on Toxigen dataset with varying model size. Moreover, as the model size becomes smaller, the model performance will decrease, and the performance gain provided by using rationales also becomes smaller, which indicates that larger student LMs can learn to generate rationales better.

Q5: Can student LMs fine-tuned with rationales actually generate high-quality rationales? We conduct a case study on the quality of rationales generated by FC-T5 using different approaches via manual check. We observe that after fine-tuning with both labels and rationales ($\mathbf{R}_{DToT}$), FC-T5 always provides responses with rich rationales. In contrast, fine-tuning with labels only makes FC-T5 overfit the binary labels and hence output 'Yes/No' answers only without rationales. In addition, without fine-tuning, FC-T5 cannot consistently generate meaningful rationales (see Table 5 in Appendix C for detailed examples).

Conclusions and Limitations

In this work, we propose an end-to-end approach to bootstrapping and distilling LLMs for toxic content detection, where a novel prompting method named DToT is designed to enhance LLMs' detection performance and extract better rationales, and smaller LMs are fine-tuned with both labels and DToT-extracted rationales. Our evaluation results on four datasets consistently demonstrate the effectiveness of both the proposed DToT prompting and the proposed fine-tuning method for toxic content detection.

Limitations. First, the context selector of DToT conducts greedy search at each iteration, which may not be globally optimal. Second, we use a pre-defined context-tree in our experiments for DToT, which can not dynamically change with LLMs' response.

A Experimental Parameters and Models

Parameters for DToT Prompting. We set s_l and s_h in Eq. (1) as 0 and 90 respectively for experiments with ChatGPT. Since we observe that ChatGPT will answer 'Yes' as long as the toxicity rating is above zero, which leads to high recall while relatively low precision (i.e. high false positive), we select a small value for s_l and large value for s_h to reduce the false positive rate of ChatGPT. We set s_δ in Eq. (2) as 0.9 for confidence checker in DToT prompting, since we empirically notice that setting the threshold of confidence score as 0.9 improves the confidence of generated answers and thus enhances the overall detection accuracy.

Moreover, we use a two-level context tree in our experiments (as shown in Figure 2 in Section 3.1) and the maximal terminal step T is set as 2. Note that we did not experiment with a deeper context-tree since DToT prompting with a two-level context tree can already significantly improve LLMs' detection performance with reasonable cost.

Parameters for Rationale Distillation. We set λ in Eq. (6) as 1, in order to make the predicted answers and rationales of student LMs equally important in our experiments. During rationale distillation experiments (see Section 5.2), for each dataset, we fine-tune the student LM on the training set for 5 epochs, where stochastic gradient descent with mini-batch size of 8 and learning rate of 1e-4 are used.

Models for DToT Prompting. We select ChatGPT as our *black-box* LLM since it is the most popular model with SOTA performance. Specifically, we use gpt-3.5-turbo instead of gpt-4, since gpt-3.5-turbo has much better trade-off between model performance and experimental cost. For *white-box* LLMs, we have tried different open-source LLMs including FastChat-T5 (Zheng et al. 2023), Flan-T5-XXL (Chung et al. 2022), and Falcon-40B (Almazrouei et al. 2023). We finally select FastChat-T5 since it has the best detection accuracy using prompts from Table 6.

Models for Distillation. We use FastChat-T5 (Zheng et al. 2023) as the student LM since it has the best performance after fine-tuning, compared with other models from T5 family (Chung et al. 2022) (see Table 4 in Section 5.2). We do not consider models larger than 3B during distillation experiments, due to their high fine-tuning cost while low run-time efficiency.

B Rationale Examples

Table 5 demonstrates the responses generated by FC-T5 using different approaches, where the input sentence is *"why do you have to come to America, you should stay in your own country and fight for your freedom, not come to America"*. We observe that after fine-tuning with both labels and rationales (FT+$\mathbf{R}_{DToT}$), FC-T5 can provide a response with rich rationale (see "Rationale: The sentence ..."). By contrast, fine-tuning with only labels will make FC-T5 output a 'Yes/No' answer without any rationale. Without fine-tuning, FC-T5 also fails to output a reasonable rationale that explains why this sentence contains toxic content.

Model	Method	Response
FC-T5	CoT	Yes, this sentence contains toxic content.
	FT	Yes.
	FT+$\mathbf{R}_{DToT}$	Yes, this sentence contains toxic content. Rationale: The sentence contains hate speech and biased content towards immigrants. It implies that immigrants are not welcome in America and should stay in their own country to fight for their freedom.

Table 5: Responses generated by FC-T5 using different approaches. Note we require to detect toxic content in sentence *"why do you have to come to America, you should stay in your own country and fight for your freedom, not come to America"* in the prompt.

Model	Prompt template
ChatGPT	Context: `c`. Sentence: `s`. Does this sentence contain `type(c)` content? First, answer in "Yes" or "No". Second, rate the `type(c)` level out of 100. Third, explain your rationale briefly. The output format is given below: Answer: ... `type(c)` level: .../100. Rationale: ...
FC-T5	`c`. ### Human: "`s`". Does this sentence contain `type(c)` content? Answer yes or no, and explain your answer. ### Assistant:

Table 6: Prompt templates for different LLMs, given question q: "Does sentence `s` contain toxic content?" (see Section 3.1). Note that `c` is the context, `s` is the statement, and `type(c)` is the category of context `c` (e.g. toxic, hate and violent).

C Prompt Templates

Table 6 provides the prompt templates used in our experiments, where the question q in Section 3.1 is given as "Does sentence `s` contain toxic content?". Note that for black-box LLM (ChatGPT), we explicitly ask the model to rate the toxicity level during prompting, in order to calculate the confidence score (see Section 3.1 Confidence Checker for details).

Acknowledgments

The authors would like to thank anonymous reviewers and members from Amazon for providing insightful feedback. This work is supported in part by the National Science Foundation under grant numbers 1956435 and 1901488.

References

Almazrouei, E.; Alobeidli, H.; Alshamsi, A.; Cappelli, A.; Cojocaru, R.; Debbah, M.; Goffinet, E.; Heslow, D.; Launay, J.; Malartic, Q.; Noune, B.; Pannier, B.; and Penedo, G. 2023. Falcon-40B: an open large language model with state-of-the-art performance.

Caselli, T.; Basile, V.; Mitrović, J.; and Granitzer, M. 2020. Hatebert: Retraining bert for abusive language detection in english. *arXiv preprint arXiv:2010.12472*.

Chen, W.-L.; Yen, A.-Z.; Huang, H.-H.; Wu, C.-K.; and Chen, H.-H. 2023. ZARA: Improving Few-Shot Self-Rationalization for Small Language Models. *arXiv preprint arXiv:2305.07355*.

Chung, H. W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, E.; Wang, X.; Dehghani, M.; Brahma, S.; et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Hartvigsen, T.; Gabriel, S.; Palangi, H.; Sap, M.; Ray, D.; and Kamar, E. 2022. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*.

Hsieh, C.-Y.; Li, C.-L.; Yeh, C.-K.; Nakhost, H.; Fujii, Y.; Ratner, A.; Krishna, R.; Lee, C.-Y.; and Pfister, T. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Kim, Y.; Park, S.; and Han, Y.-S. 2022. Generalizable implicit hate speech detection using contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, 6667–6679.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213.

Liu, J.; Shen, D.; Zhang, Y.; Dolan, B.; Carin, L.; and Chen, W. 2021. What Makes Good In-Context Examples for GPT-3? *arXiv preprint arXiv:2101.06804*.

Luo, H.; Chuang, Y.-S.; Gong, Y.; Zhang, T.; Kim, Y.; Wu, X.; Fox, D.; Meng, H.; and Glass, J. 2023. SAIL: Search-Augmented Instruction Learning. *arXiv preprint arXiv:2305.15225*.

Magister, L. C.; Mallinson, J.; Adamek, J.; Malmi, E.; and Severyn, A. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

OpenAI. 2023. ChatGPT: get instant answers, find creative inspiration, and learn something new. https://openai.com/chatgpt. Accessed: 2023-08-15.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Sap, M.; Gabriel, S.; Qin, L.; Jurafsky, D.; Smith, N. A.; and Choi, Y. 2019. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*.

Sun, Z.; Wang, X.; Tay, Y.; Yang, Y.; and Zhou, D. 2022. Recitation-augmented language models. *arXiv preprint arXiv:2210.01296*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Vidgen, B.; Thrush, T.; Waseem, Z.; and Kiela, D. 2020. Learning from the worst: Dynamically generated datasets to improve online hate detection. *arXiv preprint arXiv:2012.15761*.

Wang, P.; Chan, A.; Ilievski, F.; Chen, M.; and Ren, X. 2022a. Pinto: Faithful language reasoning using prompt-generated rationales. *arXiv preprint arXiv:2211.01562*.

Wang, P.; Wang, Z.; Li, Z.; Gao, Y.; Yin, B.; and Ren, X. 2023. SCOTT: Self-consistent chain-of-thought distillation. *arXiv preprint arXiv:2305.01879*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022b. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, X.; Zhu, W.; and Wang, W. Y. 2023. Large language models are implicitly topic models: Explaining and finding good demonstrations for in-context learning. *arXiv preprint arXiv:2301.11916*.

Wang, Y.-S.; and Chang, Y. 2022. Toxicity detection with generative prompt-based inference. *arXiv preprint arXiv:2205.12390*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Ye, Y.; Le, T.; and Lee, D. 2023. NoisyHate: Benchmarking Content Moderation Machine Learning Models with Human-Written Perturbations Online. *arXiv preprint arXiv:2303.10430*.

Zhang, T.; Luo, H.; Chuang, Y.-S.; Fang, W.; Gaitskell, L.; Hartvigsen, T.; Wu, X.; Fox, D.; Meng, H.; and Glass, J. 2023. Interpretable unified language checking. *arXiv preprint arXiv:2304.03728*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. arXiv:2306.05685.